

**Program 1**

Write a program to generate prime factors of a number input by the user.

**Sample Data**

Input the number to be prime factorised       :196

The Factors are       : 2 x 2 x 7 x 7 x 1

**Solution**-

import java.util.\*;

class primeFactors

{    int a=0;int factors[]= new int[1000];

        public static void mainPrimeFactors (String args[])

    {     System.out.print("Input the number to be prime factorised\t:");

     Scanner s=new Scanner(System.in);

     primeFactors p=new primeFactors();

     p.primeFactorise(s.nextInt());

     int x=0;

     System.out.print("The Factors are\t: ");

     while (p.factors[x]!=0)

        {               if (p.factors[x+1]!=0)

            System.out.print (p.factors[x++]+" x ");

            else

            System.out.print (p.factors[x++]+" x 1");        }    }

    public int nextPrime()

//prints the next prime number everytime it is invoked

    {        a++;

        for (;;a++)        {

            int flag=0;

            for (int b=1;b<=a;b++)

                if (a%b==0)

                    flag++;

            if (flag==2)return(a);        }    }

    public void printPrime(int n)

// prints all prime numbers less than n

    {        for(int x=0;x<n+1;x++)System.out.println(nextPrime() + ",");    }

 public void resetNextPrime()

//resets the method nextPrime()

    {        int a=0;    }

    public void primeFactorise(int n)

//prints and stores all prime factors of n in an array

    {        primeFactors pr=new primeFactors();

        if (n==1)factors[0]=1;

        int i=0;

        while (n!=1)

        {            int p=pr.nextPrime();

            while (n%p==0)

            {                n/=p;

                factors[i++]=p;            }        }    }   }

**Outputs**-

Input the number to be prime factorised       :196

The Factors are       : 2 x 2 x 7 x 7 x 1

Input the number to be prime factorised       :219

The Factors are       : 3 x 73 x 1

**Program 2**

Write a program to inherit the class used in question 1 and use it to check weather a number (input by user) is a Smith number or not.

A Smith number is a composite number, the sum of whose digits is the sum of

the digits of its prime factors obtained as a result of prime

factorization. The first few such numbers are 4, 22, 27, 58, 85,94, 121 …

**Sample data-**

Input the number to be checked           :666

Smith

Input the number to be checked           :786

Not Smith

**Solution**-

import java.util.\*;

class SmithNumbers extends primeFactors

{

static SmithNumbers obj=new SmithNumbers();

public static void mainSmith(String args[])

{

Scanner s=new Scanner(System.in);

System.out.print ("Input the number to be checked\t:");

int input=s.nextInt();

if(obj.isSmith(input))

System.out.println ("Smith");

else System.out.println ("Not Smith");

}

Boolean isSmith(int n)

{   int i=0;

    int SODf=0;

    primeFactorise(n);

    int SODn=sumOfDigits(n);

    while(i<factors.length)

    SODf+=sumOfDigits(factors[i++]);

    if (SODf==SODn)

    return true;

    else return false;

}

int sumOfDigits(int in)

{int sum=0;

while (in>0)

{

sum+=in%10;

in/=10;

}

return sum;

}}

**Outputs**-

Input the number to be checked           :666

Smith

Input the number to be checked           :256

Not Smith

**Program 3**

Write a program to find the longest string in a sentence (input by user) assuming that every word is terminated by a white space and there is only one longest word.

**Sample data-**

Input a sentence here       : Mary Had A Little Lamb

The longest word in this sentence is     : Little

**Solution**-

import java.util.\*;

class Longest

{

    public String longest(String x)

    {

        StringTokenizer s=new StringTokenizer(x);

        String a[]=new String[s.countTokens()];

        int counter=0;

        while(s.hasMoreTokens())

            a[counter++]=s.nextToken();

        //Bubble sort

        for(int i=0;i<a.length;i++)

            for (int j=0;j<a.length-1;j++)

                if (a[j].length()<a[j+1].length())

                {  String temp=a[j];

                    a[j]=a[j+1];

                    a[j+1]=temp;        }

        return a[0];    }

    public static void main (String args[])

    {        Longest Longest=new Longest();

        Scanner s = new Scanner(System.in);

        System.out.print("Input a sentence here\t: ");

        System.out.print ("The longest word in this sentence is\t: "+Longest.longest(s.nextLine()));    }            }

**Outputs**-

Input a sentence here       : Mary Had A Little Lamb

The longest word in this sentence is     : Little

Input a sentence here       : All that glisters is not gold.

The longest word in this sentence is     : glisters

**Program 4**

Write a program to check weather a number(input by user) is an armstrong number or not using RECURSIVE TECHNIQUE.

**Sample Data-**

Input The Number :156

Not Armstrong

Input The Number :153

Armstrong

**Solution-**

import java.io.\*;

class ArmstrongRecursive

{

    public static void main(String args[])throws Exception

    {

        InputStreamReader in=new InputStreamReader(System.in);

        BufferedReader buf=new BufferedReader(in);

        System.out.print("Input The Number\t:");

        int a=Integer.parseInt(buf.readLine());

        armstrong ob1=new armstrong();

        if ((ob1.wow(a)==a)&&(a!=0))

            System.out.println ("Armstrong");

 else

 System.out.println ("Not Armstrong");

    }

}

class armstrong

{int b=0;

 public int wow (int a)

    {        if (a!=0)

        {            b=a;

            b/=10;

            return ((a%10)\*(a%10)\*(a%10))+wow(b);        }

        else

            return 0;    }  }

**Output-**

Input The Number :156

Not Armstrong

Input The Number :256

Not Armstrong

Input The Number :153

Armstrong

**Program 5**

A magic number is a number in which the eventual sum of digits of the number is equal to 1.

For example, 172=1+7+2=10+1+0=1

Then 172 is a magic number.

Design a class Magic to check if a given number is a magic number. Some of the members of the class are given below:

Class name               :                       Magic

Data members/instance variables:

n                                  :   stores the number

Member functions:

Magic()                      :  constructor to assign 0 to n

void getnum(int nn)  :   to assign the parameter value to the number,

n=nn

int Sum\_of\_digits(int) : returns the sum of the digits of a number

void isMagic()          :   checks if the given number is a magic number

by calling the function Sum\_of\_digits(int) and

displays appropriate message.

Specify the class Magic giving details of the constructor, void getnum(int), int

Sum\_of\_digits(int) and void isMagic().

**Solution-**

import java.io.\*;

public class Magic

{long n;

    Magic()

    {        n=0;    }

    void getnum (long nn)

    {        n=nn;    }

    long sumOfDigits(long a)

    {        long s=0;

        while (a!=0)

        {

            s+=a%10;

            a/=10;

        }return (s);

    }

    void isMagic()

    {  long sum=0;

        sum=sumOfDigits (n);

        while (sum>9)

        {

            sum=sumOfDigits (sum);

        }

        System.out.println ("In a magic number, the eventual sum of digits is 1.");

        System.out.println ("Since, the eventual sum of digits of the number "+n+" is "+sum+",");

        if (sum==1)

            System.out.println (" therefore, it is a magic number :-)");

        else

            System.out.println (" therefore it is not a magic number :-(");

    }

    public static void main(String args[])throws IOException

    {   long num=0;

        BufferedReader buf=new BufferedReader(new InputStreamReader(System.in));

        System.out.println("This program checks wheather a given number is magic number or not.");

        System.out.print("Input the number to be tested here-->");

        try

        {

            num=Long.parseLong(buf.readLine());

        }

        catch (java.lang.NumberFormatException ex)

        {

            System.out.println ("Invalid Entry");

            System.out.println ("Try Again");

            System.out.print("Dear User,\n\tPlease input the number to be tested here-->");

            num=Long.parseLong(buf.readLine());

        }

        Magic M=new Magic();

        M.getnum(num);

        M.isMagic();

    }

}

**Output**

This program checks wheather a given number is magic number or not.

Input the number to be tested here-->256

In a magic number, the eventual sum of digits is 1.

Since, the eventual sum of digits of the number 256 is 4,

 therefore it is not a magic number :-(

**Program 6**

A Transpose of an array is obtained by interchanging the elements of the rows and

columns.A class Transarray contains a two dimensional integer array of order [mxn]. The maximum value possible for both m and n is 20. Design a class Transarray to find the transpose of a given matrix. The details of the members of the class are given below:

Class name                                       :           Transarray

Data members/instance variables:

arr[][]                      :          stores the matrix elements

m                           :           integer to store the number of rows

n                            :           integer to store the number of columns

Member functions:

Transarray()   :        default constructor

Transarrayi int mm, int nn)                      :           to initialize the size of the matrix,   void fillarray()                                 :           to enter the elements of the matrix

void transpose(Transarray A)     :           to find the transpose of a given matrix.

void disparray()                              :           displays the array in a matrix form

**Solution**

import java.io.\*;

public class Transarray

{

int arr[][]=new int[20][20];

int m,n;

Transarray()

{}

Transarray(int mm,int nn)

{m=mm;

n=nn;}

void fillarray() throws IOException

{BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter the elements");

for(int i=0;i<m;i++)

{for(int j=0;j<n;j++)

{System.out.print ("Position ["+i+"],["+j+"]\t: ");

arr[i][j]=Integer.parseInt(br.readLine());}}}

void transpose(Transarray A)

{m=A.n;

n=A.m;

System.out.println ("The transpose of your array is");

for(int i=0;i<m;i++)

{for(int j=0;j<n;j++)

{arr[i][j]=A.arr[j][i];}}}

void disparray()

{for(int i=0;i<m;i++){

for(int j=0;j<n;j++)

{System.out.print(arr[i][j]+"\t");}

System.out.println();}}

public static void main()

{    try{

    Transarray T1=new Transarray(5,5);

    Transarray T2=new Transarray();

    T1.fillarray();

    T2.transpose (T1);

    T2.disparray();

      }

      catch(Exception e)

      {    System.out.println(e);        }}}

**Output**

Enter the elements

Position [0],[0]        : 12

Position [0],[1]        : 23

Position [0],[2]        : 34

.

.

.

The transpose of your array is

12       67       12       67       21

23       78       23       78       32

34       89       34       89       43

45       90       45       90       54

56       1          56       0          65

**Program 7**

A library issues books on rental basis at a 2% charge on the cost price of the

bookper day. As per the rules of the library, a book can be retained for 7 days

without any fine. If the book is returned after 7 days, a fine will also be charged for

the excess days as per the chart given below:

Number of excess days                                         Fine per day(Rs.)

1 to 5                                                                                                 2.00

6 to 10                                                                                   3.00

above 10 days                                                                     5.00

Design a class Library and another class Compute to perform the task. The details

of the two classes are given below:

Class name                                       :          Library

Data members/instance variables:

name                                         :           name of the book

author                                        :           author of the book

p                                                :          price of the book in decimals

Member functions:

Library(…)    :           parameterized constructor to assign values to data members.

void show()    :           display the book detils.

Class name                                       :          Compute

Data members/instance variables:

d      :           number of days taken in returning the book to store the fine

f       :          to store the fine

Member functions:

Compute(…) :          parameterized constructor to assign values to data members of both the classes.

void fine()      :          calculates the fine for the excess days

void display()  :          display the book details along with the number of days, fine and the total amount to be paid. Total amount is calculated as:

(2% of price of book\*total no. of days)+fine.

Specify the class Library giving details of the constructors and void show(). Using the concept of Inheritance, specify the class Compute giving details of constructor,void fine() and void display() function.

**Solution**

import java.io.\*;

import java.util.\*;

class Library

{

    String name,author;

    float price;

    Library(String name,String author,float price)

    {

        this.name=name;

        this.author=author;

        this.price=price;

    }

    void show()

    {

        System.out.println("Book Name\t: "+name);

        System.out.println("Book Author\t: "+author);

        System.out.println("Book Price\t: Rs."+price+"/-");

    }

}

public class Compute extends Library

{

    int d;

    float fine;

    Compute(String name,String author,float price,int d)

    {

        super(name,author,price);

        this.d=d;

        fine=0;

    }

    void fine()

    {

        if(d>=1 && d<=5)

            fine=2f\*d;

        else if(d>=6 && d<=10)

            fine=5\*2f+3f\*(d-5);

        else

            fine=5\*2f+5\*3f+5f\*(d-10);

    }

    void display()

    {

        show();

        System.out.println("No. of days\t:"+d+" days");

        System.out.println("Fine\t: Rs."+fine+"/-");

        System.out.println("Total amount\t: Rs."+((.02f\*price\*d)+fine)+"/-");

    }

    public static void main(String args[])throws IOException

    {

        Scanner s=new Scanner(System.in);

        System.out.print ("Input Book Name\t: ");

        String name=s.nextLine();

        System.out.print("Input Author\t: ");

        String author=s.nextLine();

        System.out.print("Input Price\t: ");

        float price=s.nextFloat();

        System.out.print("Input Number of days\t: ");

        int d=s.nextInt();

        System.out.print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

        Compute ob=new Compute(name,author,price,d);

        ob.fine();

        ob.display();

    }

}

**Output**

Input Book Name   : A Christmas Carol

Input Author           : Charles Dickens

Input Price   : 45

Input Number of days      : 19

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Book Name  : A Christmas Carol

Book Author            : Charles Dickens

Book Price    : Rs.45.0/-

No. of days   :19 days

Fine    : Rs.70.0/-

Total amount           : Rs.87.1/-

**Program 8**

To input a string (precoded), coded by replacing each character with the character which comes 'n' characters after it,  and decode it using an input defining the number of characters to shift by.

Sample data-

Enter the encoded string             :           FQNJSX%FWJ%HTRNSL%&&

How many characters to shift    :           -5

Entered Encoded message is       :           FQNJSX%FWJ%HTRNSL%&&

Decode message is                         :           ALIENS ARE COMING !!

**Solution**

import java.io.\*;

class decode

{

    public static void main(String args[])throws IOException

    {

        BufferedReader in=new BufferedReader(new InputStreamReader(System.in));

        System.out.print("Enter the encoded string\t:\t");                 // input

        String enc;

        enc=in.readLine();

        String dec="";

        System.out.print("How many characters to shift\t:\t");// input shift

        int sh=0;

        sh=Integer.parseInt(in.readLine());

        enc=enc.toUpperCase();

        System.out.print("Entered Encoded message is\t:\t" + enc);

        int l=enc.length();

        for(int i=0;i<l;i++)

        {                                                   //  loop decodes the message

            int k=enc.charAt(i),n=0;

            k+=sh;

            for(;;)

            {

                if(k>90)                 // check for capitals

                {

                    n=k-90; // change character-wise

                    k=65+n-1;

                }

                else

                    break;

            }

            if(i+1!=l)

            {

                int k1=enc.charAt(i+1),n1=0;

                k1+=sh;

                for(;;)

                {

                    if(k1>90)

                    {

                        n1=k1-90;                // convert

                        k1=65+n1-1;

                    }

                    else

                        break;

                }

                if(k==81 && k1==81)

                {

                    dec=dec+" ";

                    i++;                     // counter increase

                    continue;          // continue loop

                }

            }

            dec+=(char)(k);      // counter update

        }

        System.out.print("\nDecode message is\t\t:\t"+dec);

    }

}

**Output**

Enter the encoded string :           This is coded

How many characters to shift    :           -2

Entered Encoded message is       :           THIS IS CODED

Decode message is             :           RFGQ‑GQ‑AMBCB

Enter the encoded string :           RFGQ‑GQ‑AMBCB

How many characters to shift    :           2

Entered Encoded message is       :           RFGQ‑GQ‑AMBCB

Decode message is             :           THIS IS CODED

**Program 9**

A bank intends to design a program to display the denomination of an

input amount, upto 5 digits. The available denomination with the bank are

of rupees 1000,500,100,50,20,10,5,2 and 1.

Design a program to accept the amount from the user and display the

break-up in descending order of denominations. (i,e preference should

be given to the highest denomination available) along with the total

number of notes. [Note: only the denomination used should be displayed].

Also print the amount in words according to the digits.

Example 1:

INPUT: 14836

OUTPUT: ONE FOUR EIGHT THREE SIX

DENOMINATION:

1000 X 14 =14000

500  X 1  =500

100  X 3  =300

50   X 1  =50

5    X 1  =5

1    X 1  =1

**Solution-**

import java.io.\*;

class Bank

{

    int rev=0,amount,dummy;

    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

    public void intake() throws IOException

    {

        System.out.println("Enter the Amount:");

        amount=Integer.parseInt(br.readLine());

        if(amount >99999)

        {

            System.out.println("Invalid Amount...");

            return;

        }

        dummy=amount;

        while(dummy >0)

        {

            rev=rev\*10+dummy%10;

            dummy=dummy/10;

        }

        System.out.print("Amount in words :");

        while(rev >0)

        {

            switch(rev%10)

            {

                case 0:

                System.out.print(" ZERO");

                break;

                case 1:

                System.out.print(" ONE");

                break;

                case 2:

                System.out.print(" TWO");

                break;

                case 3:

                System.out.print(" THREE");

                break;

                case 4:

                System.out.print(" FOUR");

                break;

                case 5:

                System.out.print(" FIVE");

                break;

                case 6:

                System.out.print(" SIX");

                break;

                case 7:

                System.out.print(" SEVEN");

                break;

                case 8:

                System.out.print(" EIGHT");

                break;

                case 9:

                System.out.print(" NINE");

                break;

            }

            rev=rev/10;

        }

        System.out.println("\nDENOMINATORS:\n");

        rev=amount/1000;

        if(rev!=0)

            System.out.println("1000 X " + rev + " = " + rev\*1000);

        amount=amount%1000;

        rev=amount/500;

        if(rev!=0)

            System.out.println("500 X " + rev + " = " + rev\*500);

        amount=amount%500;

        rev=amount/100;

        if(rev!=0)

            System.out.println("100 X " + rev + " = " + rev\*100);

        amount=amount%100;

        rev=amount/50;

        if(rev!=0)

            System.out.println("50 X " + rev + " = " + rev\*50);

        amount=amount%50;

        rev=amount/20;

        if(rev!=0)

            System.out.println("20 X " + rev + " = " + rev\*20);

        amount=amount%20;

        rev=amount/10;

        if(rev!=0)

            System.out.println("10 X " + rev + " = " + rev\*10);

        amount=amount%10;

        rev=amount/5;

        if(rev!=0)

            System.out.println("5 X " + rev + " = " + rev\*5);

        amount=amount%5;

        rev=amount/2;

        if(rev!=0)

            System.out.println("2 X " + rev + " = " + rev\*2);

        amount=amount%2;

        rev=amount/1;

        if(rev!=0)

            System.out.println("1 X " + rev + " = " + rev\*1);

    }

}

**Output**

Enter the Amount:

25648

Amount in words : TWO FIVE SIX FOUR EIGHT

DENOMINATORS:

1000 X 25 = 25000

500 X 1 = 500

100 X 1 = 100

20 X 2 = 40

5 X 1 = 5

2 X 1 = 2

1 X 1 = 1

**Program 10**

Given the two positive integers p and q, where p < q. Write a program to determine how many kaprekar numbers are there in the range between 'p' and 'q' (both inclusive) and output them.

About 'kaprekar' number:

A positive whole number 'n' that has 'd' number of digits is squared and split into

2 pieces, a right hand piece that has 'd' digits and a left hand piece that has

remaining 'd' or 'd-1' digits. If sum of the pieces is equal to the number then

it's a kaprekar number.

SAMPLE DATA:

INPUT:

p=1

Q=1000

OUTPUT:

THE KAPREKAR NUMBERS ARE:

1,9,45,55,99,297,999

FREQUENCY OF KAPREKAR NUMBERS IS:7

**Solution**

import java.io.\*;

class karpekar

{    int i,p,q,c=0;

    int num;

    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

    public void take() throws IOException

    {        System.out.println("Enter the Lower Range:");

        p=Integer.parseInt(br.readLine());

        System.out.println("Enter the Upper Range:");

        q=Integer.parseInt(br.readLine());

        if(p >=q)

        {    System.out.println("Wrong Entry...");

            return;        }

        System.out.println("THE KAPREKAR NUMBERS ARE:");

        for(i=p;i<=q;i++)

        {            show(i);        }

        System.out.println("\nFREQUENCY OF KAPREKAR NUMBERS IS:"+c);   }

    public void show(int x)

    {   int digit,rev=0;int no;num=x\*x;digit=0;no=x;

        while(no >0)

        {    digit++;

            no=no/10;        }

        no=num;

        while(digit > 0)

        {            rev=rev\*10+no%10;

            no=no/10;

            digit--;        }

        // 'rev' holds the right part in reverse order and 'no' holds the left part

        rev=reverse(rev);

        if((rev+no)==x)

        {   System.out.print("  "+x);

            c++;        }    }

    private int reverse(int n)

    {        int r=0;

        while(n > 0)

        {   r=r\*10+n%10;

            n=n/10;        }

        return r;    }}

**Output**

Enter the Lower Range:

1

Enter the Upper Range:

500

THE KAPREKAR NUMBERS ARE:

  1  9  45  55  99  297

FREQUENCY OF KAPREKAR NUMBERS IS:6

**Program 11**

Input a paragraph containing ‘n’ number of sentences where (1<=n<=4). The words are to be separated with single blank space and are in upper case.

A sentence may be terminated either with a full stop (.) or question mark (?).

Perform the followings:

(i) Enter the number of sentences, if it exceeds the limit show a message.

(ii) Find the number of words in the paragraph

(iii) Display the words in ascending order with frequency.

**Solution**

import java.io.\*;

import java.util.\*;

class Sentences

{static String s,str,sarr[],strarr[];

static StringTokenizer st;

static int i,j,n,c,index=0,fre[],index1=0;

static BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

public static void take() throws IOException

{System.out.println("Enter the Number of sentences:");

n=Integer.parseInt(br.readLine());

if(n< 1 || n >4)

{ System.out.println("Wrong Input");

return;}

System.out.println("Enter the Paragraph:");

str=br.readLine();

st=new StringTokenizer(str,",.? ");

n=st.countTokens();

System.out.println("Number of Words in the paragraph="+n);

sarr=new String[n];

strarr=new String[n];

fre=new int[n];

while(st.hasMoreTokens())

{sarr[index++]=st.nextToken();}

for(i=0;i< index-1;i++)

{for(j=i+1;j< index;j++)

{if(sarr[i].compareTo(sarr[j]) > 0)

{s=sarr[i];

sarr[i]=sarr[j];

sarr[j]=s;}}}

c=1;

s=sarr[0];

for(i=1;i< index;i++)

{if(!s.equals(sarr[i]))

{strarr[index1]=s;

fre[index1++]=c;

c=1;

s=sarr[i];}

else

c++;}

strarr[index1]=s;

fre[index1++]=c;

for(i=0;i< index1-1;i++)

{for(j=i+1;j< index1;j++)

{if(fre[i] > fre[j])

{n=fre[i];

fre[i]=fre[j];

fre[j]=n;

s= strarr[i];

strarr[i]=strarr[j];

strarr[j]=s;}}}

System.out.println("WORD\t\t\tFREQUENCY");

for(i=0;i< index1;i++)

System.out.println(strarr[i]+"\t\t\t"+fre[i]);}}

**Output**

Enter the Number of sentences:

3

Enter the Paragraph:

Democracy is of the people. It is for the people. And it is by the people.

Number of Words in the paragraph=16

WORD                                   FREQUENCY

And                                                    1

Democracy                                      1

It                                                         1

by                                                       1

for                                                      1

it                                                          1

of                                                        1

is                                                         3

people                                                           3

the                                                      3

**Program 12**

Write a menu driven program to output the value of nCr or nPr (as in permutation and combination)

as chosen by the user.

**Solution**

import java.util.\*;

class PnC   //based on permutation andcombination

{

    int fact=1,F=0,P=0,C=0;

    int factorial (int num)

    {

        if (num>0)

        {

            fact\*=num;

            return factorial (num-1);

        }

        else {

            F=fact;

            fact=1;

            return F;

        }

    }

    int combination(int n,int r)

    {

        System.out.print ("The value of nCr is\t:");

        C=factorial (n)/(factorial(n-r)\*factorial (r));

        System.out.println (C);

        return C;

    }

    int permutation(int n,int r)

    {

        System.out.print ("The value of nPr is\t:");

        P=factorial (n)/factorial(n-r);

        System.out.println (P);

        return P;

    }

    public static void main (String args[])throws Exception

    {PnC o=new PnC();

        Scanner s=new Scanner(System.in);

        int n,r;

        String cp=new String("");

        System.out.println ("Input the value of n");

        n=s.nextInt();

        System.out.println ("Input the value of r");

        r=s.nextInt();

        System.out.println ("Input C for combination or input P for permutation");

        cp=s.next();

        if (cp.equalsIgnoreCase("C"))

            o.combination(n,r);

        else if (cp.equalsIgnoreCase("P"))

            o.permutation(n,r);

        else

            System.out.println ("Invalid Input \n"+cp+" is not a valid input\nInput  C  for combination or input P for permutation");

    }

}

**Output**

Input the value of n

8

Input the value of r

2

Input C for combination or input P for permutation

C

The value of nCr is      :28

Input the value of n

10

Input the value of r

4

Input C for combination or input P for permutation

P

The value of nPr is      :5040

**Program 13**

Input a sentence from the user and count the number of times an and and are present in the sentence. Design a class Frequency to perform the above task.

Some of the members of the class are given below-

|  |  |
| --- | --- |
| **Class Name** | **Frequency** |
| **Data Members/Instance Variables:** |  |
| Text | Stores the sentence. |
| Countand | Stores the frequency of ''and''. |
| countan | Stores the frequency of ''an''. |
| Len | Stores the length of the string. |
| **Member Functions:** |  |
| Frequency() | Constructor to initiaize data members . |
| void accept(String n) | To assign the value of 'n' to 'text' where the value of parameter should be in lower case. |
| void checkandfreq() | To count the frequency of the word "and". |
| void checkanfreq() | To count the frequency of the word "an". |
| void display() | Displays the respective frequencies of "an" and "and" giving appropriate messages. |

Specify the class Frequency, giving details of the constructor, void accept(String), void checkandfreq() , void checkanfreq() and void display(). Also define a main() function to create an object and call the functions accordingly to enable the task.

**Solution**

import java.util.\*;

class Frequency

{

    String text;

    int countan,countand,len;

    Frequency()

    {

        countan=0;

        countand=0;

        len=0;

        text="";

    }

    void accept(String n)

    {

        text=n.toLowerCase();

    }

    void checkandfreq()

    {

        StringTokenizer s=new StringTokenizer(text);

        while(s.hasMoreTokens())

        {

            if(s.nextToken().equalsIgnoreCase("and"))

                countand++;

        }

    }

    void checkanfreq()

    {

        StringTokenizer s=new StringTokenizer(text);

        while(s.hasMoreTokens())

        {

            if(s.nextToken().equalsIgnoreCase("an"))

                countan++;

        }

    }

    void display()

    {

        System.out.println ("Frequency of an\t: "+countan+"\nFrequency of and\t: "+countand);

    }

    public static void main (String args[])

    {

        Frequency obj=new Frequency();

        Scanner s=new Scanner (System.in);

        System.out.print ("Enter the string\t: ");

        try{obj.accept(s.nextLine());

            obj.checkanfreq();

            obj.checkandfreq();

            obj.display();

        }

        catch (Exception e)

        {

            System.out.println (e);        }    }}

**Output**

Enter the string           : I and You and an antennae are enough

Frequency of an          : 1

Frequency of and        : 2

**Program 14**

Write a program to check whether a given number is a "Perfect Number" or not. A perfect number is the one whose factors (including 1 and excluding the number itself), add up to give the number itself. For Example-

6 is a perfect number

28 is a perfect number

496 is a perfect number

**Solution**

import java.io.\*;

class PerfectNumber

{

    int no,sum=0,i;

    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

    public void takeNumber()throws Exception

    {

        System.out.print("Enter the number:");

        no=Integer.parseInt(br.readLine());

        for(i=1;i<no;i++)

        {

            if(no%i==0)

                sum=sum+i;

        }

        if(sum==no)

            System.out.println(no + " is a perfect number");

        else

            System.out.println(no + " is not a perfect number");

    }

    public static void main(String args[])throws Exception

    {

        PerfectNumber obj=new PerfectNumber ();

        obj.takeNumber();

    }       }

**Output**

Enter the number:496

496 is a perfect number

Enter the number:28

28 is a perfect number

Enter the number:128

128 is not a perfect number

**Program 15**

Write a program to input from the user, a square matrix of the size of user's choice and check wheather the matrix is a wonderous square or not. Also print the prime numbers present in the matrix along with their row and column number in tabular form.

A wondrous square is an n by n grid which fulfils the following conditions: (i) It contains integers from 1 to n2 , Where each integer appears only once. (ii) The some of integers in any row or column must add up to 0.5×n×(n2 +1).

**Solution**

import java.io.\*;

class WonderousSquare

{

    int arr[][],arr1[];;

    int n,i,j,x=0,r,c;

    int flag;

    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

    public void take()throws Exception

    {

        System.out.println("\nEnter the size of array(row and column same):");

        n=Integer.parseInt(br.readLine().trim());

        arr=new int[n][n];

        arr1=new int[2\*n];

        for(i=0;i< n;i++)

        {

            for(j=0;j< n;j++)

            {

                System.out.println("\nEnter the value:");

                arr[i][j]=Integer.parseInt(br.readLine());

            }

        }

        System.out.println("\nThe matrix is\n");

        for(i=0;i< n;i++)

        {

            r=0;

            c=0;

            for(j=0;j< n;j++)

            {

                System.out.print(arr[i][j]+" ");

                r=r+arr[i][j];

                c=c+arr[j][i];

            }

            System.out.println();

            arr1[x]=r;

            arr1[x+n-1]=c;

            x++;

        }

        for(i=0;i< x;i++)

        {

            if(arr1[i]!= 0.5 \* n \* (n\*n + 1))

                break;

        }

        if(i==x)

            System.out.println("YES IT REPRESENTS A WONDROUS SQUARE.");

        else

            System.out.println("IT IS NOT A WONDROUS SQUARE.");

        System.out.println("PRIME\tROW\tCOLUMN");

        for(i=0;i< n;i++)

        {

            for(j=0;j< n;j++)

            {

                if(prime(arr[i][j]))

                    System.out.println(arr[i][j]+ "\t"+i+ "\t"+j);

            }

        }

    }

    private boolean prime(int no)

    {

        int index;

        for(index=2;index< no;index++)

        {

            if(no%index==0)

                break;

        }

        if(index==no)

            return true;

        else

            return false;

    }

    public static void main(String args[])throws Exception

    {

        WonderousSquare ob=new WonderousSquare();

        ob.take();

    }

}

**Output-**

Enter the size of array(row and column same) : 2

Enter the value : 1

Enter the value : 2

Enter the value : 3

Enter the value : 4

The matrix is

1          2

3          4

IT IS NOT A WONDROUS SQUARE.

            PRIME ROW    COLUMN

            2          0          1

            3          1          0

**Program 16**

Write a class Anagram to print all possible anagrams of a word input by the user. Also print the number of possible anagrams of the word.

**Solution**

import java.io.\*;

 public class Anagrams

 {

String str;

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

int counter=0;

public void take()throws Exception

 {

   System.out.println("\nEnter the word:");

   str=br.readLine();

   show("", str);

   System.out.println("Total Number of Anagrams="+counter);

 }

   public void show(String s, String str)

  {

      if(str.length()<= 1)

     {

       counter++;

       System.out.println(s+str);

     }

    else

     {

        for(int i = 0; i< str.length(); i++)

       {

          String str1 = str.substring(i, i + 1);

          String str2 = str.substring(0, i);

          String str3 = str.substring(i + 1);

          show(s + str1, str2 + str3);

        }     }   }

public static void main(String args[])throws Exception

{ Anagrams ob=new Anagrams();

ob.take();}}

**Output**

Enter the word:

pat

pat

pta

apt

atp

tpa

tap

Total Number of Anagrams=6

**Program 17**

Write a program to check wheather a number is automorphic or not. An automorphic number is the one whose square ends with the number itself.

For example, 25 is an automorphic number. Number of digits in 25 is 2 and square value of 25 is 625. We have to take 2 extreme right digits from 625 as there are 2 digits in the original entered number. Two extreme right digits of 625 is 25 which matches with original number 25. So 25 is an automorphic number. Similarly 5 is also an automorphic number.

**Solution**

import java.io.\*;

class Automorphic

{int i,n,no,sqnum,rev=0,digit=0;

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

public void getNumber() throws Exception

{System.out.println("Enter the number:");

n=Integer.parseInt(br.readLine());

no=n;

do

{digit++;

no=no/10;

} while(no!=0);

sqnum=n\*n;}

public void showResult()

{do

{rev=rev\*10+sqnum%10;

sqnum=sqnum/10;

digit--;

if(digit==0)

break;

} while(true);

rev=reverse(rev);

if(n==rev)

System.out.println(n+" is an Automorphic number");

else

System.out.println(n+" is not an Automorphic number");}

private int reverse(int n)

{int r=0;

while(n!=0)

{r=r\*10+n%10;

n=n/10;}

return r;}

public static void main(String args[])throws Exception

{Automorphic obj=new Automorphic();

obj.getNumber();

obj.showResult();}}

**Output**

Enter the number:

25

25 is an Automorphic number

Enter the number:

30

30 is not an Automorphic number

Enter the number:

5

5 is an Automorphic number

**Program 18**

Check whether a number is Special Number in JAVA.  
A number is said to be special number when the sum of factorial of its digits is equal to the number itself.  
Example- 145 is a Special Number as 1!+4!+5!=145.  
  
**Solution-**  
  
import java.util.\*;  
public class SpecialNumberCheck  
{  
    public static void main(String args[])  
    {  
        Scanner ob=new Scanner(System.in);  
        System.out.println("Enter the number to be checked.");  
        int num=ob.nextInt();  
        int sum=0;int temp=num;  
        while(temp!=0)  
        {  
            int a=temp%10;int fact=1;  
            for(int i=1;i<=a;i++)  
            {  
                fact=fact\*i;  
            }  
            sum=sum+fact;  
            temp=temp/10;  
        }  
        if(sum==num)  
        {  
            System.out.println(num+" is a Special Number.");  
        }  
        else  
        {  
            System.out.println(num+" is not a Special Number.");  
        }  
    }  
}

**Output:**

Enter the number to be checked.

145

145 is a Special Number.

**Program 19**

The co-ordinates of a point P on a two dimensional plane can be represented by P(x,y) with x as the x-coordinate and y as the y-coordinate. the coordinates of midpoint of two points P1(x1,y1) and P2(x2,y2) can be calculated as P(x,y) where

x=(x1+x2)/2 and y=(y1+y2)/2.

Design a class Point with the following details:

Class Name                                                   :           Point

Data Member/Instance Variables

x                     :           Stores the x-coordinate

y                     :           Stores the y-coordinate

Member Functions

Point ()                                                         Constructor to initialise the instance variables.

void readpoint ()                                           Accepts the coordinates x and y of a point.

Point midpoint(Point A , Po            Calculates and returns the midpoint of the two points A and B.

void displaypoint()                                         Displays the coordinates of a point.

                        Specify the class Point giving details of the constructor(), member functions voidreadpoint(), Point midpoint(Point, Point) and void displaypoint() along with the main() function to create an object and call the functions accordingly to calculate the midpoint between any two given points.

**Solution**

import java.io.\*;

class Point

{    double x,y;

    Point()

    {        x=0;

        y=0;    }

    Point (double x, double y)

    {        x=this.x;

        y=this.y;    }

    void readpoint()throws Exception

    {        BufferedReader buf=new BufferedReader (new InputStreamReader(System.in));

        System.out.print("Enter the values of x and y\nvalue of x\t:");

        x=Double.parseDouble(buf.readLine ());

        System.out.print("value of y\t:");

        y=Double.parseDouble(buf.readLine ());    }

    Point midpoint(Point A,Point B)

    {        x=(A.x+B.x)/2;

        y=(A.y+B.y)/2;

        Point ret=new Point (x,y);

        return ret;    }

    void displaypoint()

    {        System.out.println ("("+x+","+y+")");    }

    public static void main()

    {try        {

            Point point1=new Point();

            Point point2=new Point();

            point1.readpoint();

            point2.readpoint();

            Point x=point1.midpoint(point1,point2);

            point1.displaypoint();        }

        catch (Exception e)

        {            System.out.println ("Exception Occurred\n"+e);        }    }}

**Output**

Point.main();

Enter the values of x and y

value of x      :2

value of y      :3

Enter the values of x and y

value of x      :4

value of y      :5

(3.0,4.0)

**Program 20**

Write a program to input a string and print the following pattern :

 String =  holiday

\_

h\_y

ho\_ay

hol\_day

holi\_iday

holid\_liday

holida\_oliday

holiday\_holiday

holida\_oliday

holid\_liday

holi\_iday

hol\_day

ho\_ay

h\_y

\_

**Solution**

import java.io.\*;

class pattern

{    public static void main(String args[])throws IOException

    {   BufferedReader in=new BufferedReader(new InputStreamReader(System.in));

        System.out.println("Enter the string");   // input

        String S="";

        S=in.readLine();

        int n=S.length();              // function returns length of string input

        int a,i,j,k=-1,s=n-1,m=n;

        System.out.println();

        for(a=0;a<=n;a++)                       // loop generates half pattern

        {      for(i=0;i<=s;i++)       // nested looping

                System.out.print(" ");

            s--;                               // update counter

            for(i=0;i<=k;i++)          //nested looping

                System.out.print(S.charAt(i));

            k++;                             // update counter

            System.out.print("\_");

            for(j=m;j<=n-1;j++)     // nested looping

                System.out.print(S.charAt(j));

            m--;                             // update counter

            System.out.println();  }// come to next line

         s=0;k=n-2;m=1;

        for(a=1;a<=n;a++)

        {            for(i=0;i<=s;i++) // nested loop

                System.out.print(" ");

            s++;                              // counter

            for(i=0;i<=k;i++)          // nested loop

                System.out.print(S.charAt(i));

            k--;                               // counter

            System.out.print("\_");

            for(j=m;j<=n-1;j++)     // nested loop

                System.out.print(S.charAt(j));

            m++;                // counter

            System.out.println();}}}           // come to next line

**OUTPUT :**

Enter the string

cricket

\_

c\_t

cr\_et

cri\_ket

cric\_cket

crick\_icket

cricke\_ricket

cricket\_cricket

cricke\_ricket

crick\_icket

cric\_cket

cri\_ket

cr\_et

c\_t

\_

**Program 21**

Write a program to input a string and to print it in lexicographical order of characters. The program should also ask the user weather he wants to  try again.

**Sample Data**

Enter the string here--> wonderful

The string in alphabetical order is  :  deflnoruw

Try Again?

No

**Solution**

import java.io.\*;

class alphabet

{    public static void main(String args[])throws Exception

    {  byte z=1;

        while(z==1)

        {   alphabet o=new alphabet();

            BufferedReader s=new BufferedReader(new InputStreamReader(System.in));

            System.out.print("Enter the string here--> ");

            String S=s.readLine();

            System.out.println("The string in alphabetical order is\t: "+o.alpha(S));

            System.out.println("Try Again?");

            if (s.readLine().equalsIgnoreCase("yes"))

            {z=1;}

            else z=0;        }    }

    public String alpha(String S)

    {   String newString="";

        S=S.toLowerCase();

        char s[]=new char[S.length()];

        for (int i=0;i<S.length();i++)

            s[i]=S.charAt(i);

        for (int j=0;j<s.length-1;j++)

            for (int i=0;i<s.length-1;i++)

            {    if(s[i]>s[i+1])

                {   char temp=s[i];

                    s[i]=s[i+1];

                    s[i+1]=temp;                }        }

        for (int i=0;i<s.length;i++)

            newString+=(s[i]);

        return newString;    }}

**Output**

Enter the string here--> qwertyuiopasdfghjklzxcvbnm

The string in alphabetical order is     : abcdefghijklmnopqrstuvwxyz

Try Again?

yes

Enter the string here--> Gaurav

The string in alphabetical order is  :  aagruv

Try Again?

No

**Program 22**

Write a program to input a string and output the reverse of the string input by the user.

**Sample Data**

Please enter the string to be reversed           :

Anthony Gonzalves

The reversed string is :sevlaznoG ynohtnA

**Solution**

import java.io.\*;

class StringReverser

{

    String s,s1;

    String reverseString (String St)

    {

        s=St;

        s1="";

        int l=s.length();

        int pos=l-1;

        while (pos>(-1))

        {

            s1=s1+(s.charAt(pos));

            pos--;

        }

        return s1;

    }

    public static void main(String args[])throws Exception

    {

        BufferedReader buf=new BufferedReader(new InputStreamReader (System.in));

        System.out.println ("Please enter the string to be reversed\t:");

        String in=buf.readLine();

        StringReverser sr=new StringReverser ();

        System.out.println ("The reversed string is\t:"+sr.reverseString (in));

    }

}

**Output**

Please enter the string to be reversed           :

Anthony Gonzalves

The reversed string is :sevlaznoG ynohtnA

Please enter the string to be reversed           :

Madam is Malayalam

The reversed string is :malayalaM si madaM 

**Program 23**

Write a program to inherit the class used in question 22 and to use it to check weather a string is pallindrome or not.

**Sample Data**-

Enter the string to be checked           :

Malayalam

Pallindrome

Enter the string to be checked           :

Hello World

Not Pallindrome

**Solution**

import java.io.\*;

class StringPallindrome extends StringReverser

{

    StringReverser rev=new StringReverser();

    public void pallindrome(String s)

    {

        if(rev.reverseString(s).equalsIgnoreCase(s))System.out.println ("Pallindrome");

        else System.out.println ("Not Pallindrome");

    }

    public static void main(String args[])throws Exception

    {

        BufferedReader buf=new BufferedReader(new InputStreamReader (System.in));

        System.out.println ("Enter the string to be checked\t:");

        StringPallindrome pal=new StringPallindrome();

        pal.pallindrome(buf.readLine());

    }

}

**Output**

Enter the string to be checked           :

Titanic

Not Pallindrome

Enter the string to be checked           :

Tet a tet

Pallindrome

**Program 24**

Write a program in java to input elements in an array and sort it using bubble sort.

**Solution-**

import java.util.Scanner;

 class BubbleSort {

public static void main(String []args) {

int n, c, d, swap;

Scanner in = new Scanner(System.in);

System.out.println("Input number of integers to sort");

n = in.nextInt();

int array[] = new int[n];

System.out.println("Enter " + n + " integers");

for (c = 0; c < n; c++)

array[c] = in.nextInt();

for (c = 0; c < ( n - 1 ); c++) {

for (d = 0; d < n - c - 1; d++) {

if (array[d] > array[d+1]) */\* For descending order use < \*/*

{

swap = array[d];

array[d] = array[d+1];

array[d+1] = swap;

}

}

}

System.out.println("Sorted list of numbers");

for (c = 0; c < n; c++)

System.out.println(array[c]);

}

}

**Output-**

Input number of integers to sort

5

Enter 5 integers

3

8

2

1

9

Sorted list of numbers

1

2

3

8

9

**Program 25**

Write a program in java to input elements in an array and sort it using selection sort.

**Solution-**

import java.util.Scanner;

 public class Sort\_Selection {

int numbers[] ;

int n;

Scanner scan = new Scanner(System.in);

void getNumbers() {

System.out.println("Selection Sort");

System.out.println("\nEnter n value :");

n = Integer.parseInt(scan.nextLine());

numbers = new int[n];

System.out.println("Enter the Numbers :");

for(int i=0; i<n; i++) {

numbers[i] = Integer.parseInt(scan.nextLine());

}

}

void SelectionSort() {

System.out.println("\nBefore Sorting :");

for(int i=0; i<n; i++) {

System.out.print(numbers[i] + " ");

}

for(int i=0; i<n; i++) {

int index\_of\_min = i;

for(int j=i; j<n; j++) {

if(numbers[index\_of\_min] > numbers[j]) {

index\_of\_min = j;

}

}

int temp = numbers[i];

numbers[i] = numbers[index\_of\_min];

numbers[index\_of\_min] = temp;

}

System.out.println("\n \nAfter Sorting");

System.out.println("\nAscending Order :");

for(int i=0; i<n; i++) {

System.out.print(" " + numbers[i]);

}

}

}

class MainClass {

public static void main(String args[]) {

Sort\_Selection obj = new Sort\_Selection();

obj.getNumbers();

obj.SelectionSort();

}

}

**Output-**

Selection Sort

Enter n value :

5

Enter the Numbers :

6

3

5

2

4

Before Sorting :

6 3 5 2 4

After Sorting

Ascending Order :

2 3 4 5 6

**Program 26**

Write a program in java to push and pop elements in a stack using array.

**Solution-**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

class Stack {

    private int top;

    private int item[];

    Stack(int size) {

        top = -1;

        item = new int[size];

    }

    void pushItem(int data) {

        if (top == item.length - 1) {

            System.out.println("Stack is Full");

        } else {

            item[++top] = data;

            System.out.println("Pushed Item :" + item[top]);

        }

    }

    int popItem() {

        if (top < 0) {

            System.out.println("Stack Underflow");

            return 0;

        } else {

            System.out.println("Pop Item : " + item[top]);

            return item[top--];

        }

    }

}

class StackExample {

    public static void main(String[] args) throws IOException {

        Stack stk = new Stack(5);

        boolean yes=true;

        int choice;

        BufferedReader is = new BufferedReader(new InputStreamReader(System.in));

        do{

            System.out.println("1).Push\n2).Pop\n3).Exit\n\nEnter Choice");

            choice = Integer.parseInt(is.readLine());

                        switch(choice)

            {

                case 1: System.out.println("Enter Push Item: ");

                        stk.pushItem(Integer.parseInt(is.readLine()));

                        break;

                case 2: stk.popItem();break;

                case 3: yes = false;break;

                default: System.out.println("Invalid Choice");

            }

        }while(yes==true);

            }

}

**Output-**

Enter Choice

1

Enter Push Item:

14

Pushed Item :14

1).Push

2).Pop

3).Exit

Enter Choice

1

Enter Push Item:

567

Pushed Item :567

1).Push

2).Pop

3).Exit

Enter Choice

2

Pop Item : 56

1).Push

2).Pop

3).Exit

Enter Choice

2

Pop Item : 789

1).Push

2).Pop

3).Exit

Enter Choice

3

**Program 27**

Write a program in java in a queue using array.

**Solution-**

import java.io.\*;

class Queue

{

int Q[]; // Array to implement Queue

int size; // Maximum size of the Queue

int front; // Index of front element

int rear; // Index of rear element

Queue(int cap) // Parameterised Constructor

{

size = cap;

Q = new int[size];

front = 0;

rear = 0;

}

void insert(int v) // Function to insert element in Queue

{

if(rear == size) // Condition for Overflow

{

System.*out*.println("OVERFLOW");

}

else

{

Q[rear] = v; // Storing value in Queue

rear = rear + 1;

}

}

public static void main(String[] args) throws IOException {

Queue q = new Queue(5);

boolean yes=true;

int choice;

BufferedReader is = new BufferedReader(new InputStreamReader(System.*in*));

do{

System.*out*.println("1).Enqueue\n2).Dequeue\n3).Exit\n\nEnter Choice");

choice = Integer.*parseInt*(is.readLine());

switch(choice)

{

case 1: System.*out*.println("Enter Item: ");

q.insert(Integer.*parseInt*(is.readLine()));

break;

case 2: q.delete();break;

case 3: yes = false;break;

default: System.*out*.println("Invalid Choice");

}

}while(yes==true);

}

void delete() // Function to delete element from Queue

{

if(front == 0 && rear == 0) // Condition for Underflow

{

System.*out*.println("UNDERFLOW");

}

else

{

int val = Q[front]; // Storing the element which will be removed

front = front + 1;

if(front == rear) // Condition for emptying the Queue

{

front = 0;

rear = 0;

}

}

}

void display() // Function for printing elements in the queue

{

if(front == 0 && rear == 0)

{

System.*out*.println("The Queue is empty");

}

else

{

System.*out*.println("The elements in the queue are : ");

for(int i=front; i<rear; i++)

{

System.*out*.println(Q[i]);

}

}

}

}

**Output-**

1).Enqueue

2).Dequeue

3).Exit

Enter Choice

1

Enter Item:

3

1).Enqueue

2).Dequeue

3).Exit

Enter Choice

1

Enter Item:

4

1).Enqueue

2).Dequeue

3).Exit

Enter Choice

2

1).Enqueue

2).Dequeue

3).Exit

Enter Choice

4

Invalid Choice

1).Enqueue

2).Dequeue

3).Exit

Enter Choice

3

**Program 28**

Write a Program in Java to input a 2-D square matrix and check whether it is a Scalar Matrix or not.

**Scalar Matrix :** A scalar matrix is a [**diagonal matrix**](http://www.guideforschool.com/1961892-java-program-to-check-for-diagonal-matrix/) in which the main diagonal () entries are all equal.

**Example:**



**Solution-**

import java.util.\*;

class ScalarMatrix

{

    public static void main(String args[])throws Exception

    {

        Scanner sc=new Scanner(System.in);

        System.out.print("Enter the size of the matrix : ");

        int m=sc.nextInt();

        int A[][]=new int[m][m];

        /\* Inputting the matrix \*/

        for(int i=0;i<m;i++)

        {

            for(int j=0;j<m;j++)

            {

                System.out.print("Enter an element : ");

                A[i][j]=sc.nextInt();

            }

        }

        /\* Printing the matrix \*/

        System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

        System.out.println("The Matrix is : ");

        for(int i=0;i<m;i++)

        {

            for(int j=0;j<m;j++)

            {

                System.out.print(A[i][j]+"\t");

            }

            System.out.println();

        }

        System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

        int p = 0, q = 0, x = A[0][0]; // 'x' is storing the 1st main diagonal element

        for(int i=0;i<m;i++)

        {

            for(int j=0;j<m;j++)

            {

                /\* Checking that the matrix is diagonal or not \*/

                if(i!=j && A[i][j]!=0) // All non-diagonal elements must be zero

                {

                    p=1;

                    break;

                }

                /\* Checking the matrix for scalarity \*/

                // All main diagonal elements must be equal to 'x' and non-zero

                if(i==j && (A[i][j]==0 || A[i][j]!=x))

                {

                    q=1;

                    break;

                }

            }

        }

        if(p==0 && q==0)

            System.out.println("The matrix is scalar");

        else

            System.out.println("The matrix is not scalar");

    }

}

**Output-**

Enter the size of the matrix : 4  
Enter an element : 5  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 5  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 5  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 5  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
The Matrix is :  
5 0 0 0  
0 5 0 0  
0 0 5 0  
0 0 0 5  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
The matrix is Scalar

**Program 29**

Write a Program in Java to input a 2-D square matrix and check whether it is a Diagonal Matrix or not.

**Diagonal Matrix :** A diagonal matrix is a matrix (usually a square matrix) in which the entries outside the main diagonal () are all zero. The diagonal entries themselves may or may not be zero (but all diagonal entries cannot be zero).

**Example:**



**Solution-**

import java.util.\*;

class DiagonalMatrix

{

    public static void main(String args[])throws Exception

    {

        Scanner sc=new Scanner(System.in);

        System.out.print("Enter the size of the matrix : ");

        int m=sc.nextInt();

        int A[][]=new int[m][m];

        /\* Inputting the matrix \*/

        for(int i=0;i<m;i++)

        {

            for(int j=0;j<m;j++)

            {

                System.out.print("Enter an element : ");

                A[i][j]=sc.nextInt();

            }

        }

        /\* Printing the matrix \*/

        System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

        System.out.println("The Matrix is : ");

        for(int i=0;i<m;i++)

        {

            for(int j=0;j<m;j++)

            {

                System.out.print(A[i][j]+"\t");

            }

            System.out.println();

        }

        System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

        int p=0, q=0;

        for(int i=0;i<m;i++)

        {

            for(int j=0;j<m;j++)

            {

                if(i!=j && A[i][j]!=0) // Checking non-diagonal elements

                {

                    p=1;

                    break;

                }

                if(i==j && A[i][j]==0) // Checking diagonal elements

                {

                    q++;

                }

            }

        }

        if(p==0 && q<m)

            System.out.println("The matrix is Diagonal");

        else

            System.out.println("The matrix is not Diagonal");

    }

}

**Output-**

Enter the size of the matrix : 4  
Enter an element : 5  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 1  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 0  
Enter an element : 7  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
The Matrix is :  
5 0 0 0  
0 1 0 0  
0 0 0 0  
0 0 0 7  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
The matrix is Diagonal

**Program 30**

Write a program in java to search an element in an array using Linear search.

**Solution-**

import java.util.Scanner;

class LinearSearch

{

public static void main(String args[])

{

int c, n, search, array[];

Scanner in = new Scanner(System.in);

System.out.println("Enter number of elements");

n = in.nextInt();

array = new int[n];

System.out.println("Enter " + n + " integers");

for (c = 0; c < n; c++)

array[c] = in.nextInt();

System.out.println("Enter value to find");

search = in.nextInt();

for (c = 0; c < n; c++)

{

if (array[c] == search) */\* Searching element is present \*/*

{

System.out.println(search + " is present at location " + (c + 1) + ".");

break;

}

}

if (c == n) */\* Searching element is absent \*/*

System.out.println(search + " is not present in array.");

}}

**Output-**

Enter number of elements

7

Enter 7 integers

5

65

656

82

1

3

4

Enter value to find

656

656 is present at location 3

**Program 31**

Write a program in java to search an element in an array using Binary search.

**Solution-**

import java.util.Scanner;

class BinarySearch

{

public static void main(String args[])

{

int c, first, last, middle, n, search, array[];

Scanner in = new Scanner(System.in);

System.out.println("Enter number of elements");

n = in.nextInt();

array = new int[n];

System.out.println("Enter " + n + " integers");

for (c = 0; c < n; c++)

array[c] = in.nextInt();

System.out.println("Enter value to find");

search = in.nextInt();

first = 0;

last = n - 1;

middle = (first + last)/2;

while( first <= last )

{

if ( array[middle] < search )

first = middle + 1;

else if ( array[middle] == search )

{

System.out.println(search + " found at location " + (middle + 1) + ".");

break;

}

else

last = middle - 1;

middle = (first + last)/2;

}

if ( first > last )

System.out.println(search + " is not present in the list.\n");

}

}

**Output-**

Enter number of elements

7

Enter 7 integers

5

65

656

82

1

3

4

Enter value to find

656

656 is present at location 3

**Program 32**

Design a class to overload a function series() as follows:

(i) double series(double n) with one double argument and returns the sum of the series.  
sum = 1/1 + 1/2 + 1/3 + … 1/n

(ii) double series(double a, double n) with two double arguments and returns the sum of the series.  
sum = 1/a2 + 4/a5 + 7/a8 + 10/a11 … to n terms … to n terms

**Solution-**

import java.io.\*;

public class SeriesOverLoaded {

    double s;     // Instance variable for sum of the series

    double i;    // Instance variable for counter

    // Constructor of the class

    SeriesOverLoaded() {

        s = 0.0;     // Initialize the variable s

    }

    // First Method of series

    double series(double n) {

        for (i = 1.0; i <= n; i++) {

            s = s + 1 / i;

        }

        return s;

    }

    // Second method of the series

    double series(double a, double n) {

        for (i = 1.0; i <= n; i += 3) {

            s = s + i / Math.pow(a, i + 1);

        }

        return s;

    }

    // Main method started

    public static void main(String args[])throws IOException {

        double a, n, s;

        SeriesOverLoaded ob = new SeriesOverLoaded();

        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

        // Fisrt series(a,n) Function call

        System.out.println("Enter n term :");

        n = Double.parseDouble(br.readLine());

        s = ob.series(n);

        System.out.println("Series=" + s);

        // Second series(a,n) Function call

        System.out.println("Enter n and a terms :");

        n = Double.parseDouble(br.readLine());

        a = Double.parseDouble(br.readLine());

        s = ob.series(a, n);

        System.out.println("Series=" + s);

    }

} // main method ended

**Output-**

Enter n term :  
10  
Series=2.9289682539682538

Enter n and a terms :  
20  
5  
Series=2.9702663809193446

**Program 33**

**Write a program in java to accept two numbers through user and check if they are twin prime numbers or not.**

A twin prime is a prime number that differs from another prime number by two. Except for the pair (2, 3), this is the smallest possible difference between two primes. Some examples of twin prime pairs are (3, 5), (5, 7), (11, 13), (17, 19), (29, 31), (41, 43) …. (821, 823), etc.

**Solution-**

import java.io.\*;

public class TwinPrimeNumber {

     public static int primechecker(int n) {

        int f = 0;

        for (int i = 2; i <= n / 2; i++) {

            if (n % i == 0) {

                f = 1;

                break;

            }

        }

        return f;

    }

    // main method begins execution of this class

    public static void main(String args[]) throws IOException{

        int num1, num2;

 BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

        System.out.print("Enter first number: ");

        // Waiting for the user input

        num1 = Integer.parseInt(br.readLine());

        System.out.print("Enter second number: ");

        // Waiting for the user input

        num2 = Integer.parseInt(br.readLine());

        // if both are prime and difference between them is 2

        if (primechecker(num1) == 0 && primechecker(num2) == 0 && Math.abs(num2 - num1) == 2) {

            System.out.println("The numbers are twin prime numbers");

        } else {

            System.out.println("The numbers are not twin prime numbers");

        }

    } // end method main

} // end class

**Output-**

Enter first number: 17  
Enter second number: 19  
The numbers are twin prime numbers

Enter first number: 11  
Enter second number: 19  
The numbers are not twin prime numbers

**Program 34**

A **Circular Prime** is a prime number that remains prime under cyclic shifts of its digits. When the leftmost digit is removed and replaced at the end of the remaining string of digits, the generated number is still prime. The process is repeated until the original number is reached again.  
A number is said to be prime if it has only two factors I and itself.

**Example:**  
131  
311  
113  
Hence, 131 is a circular prime.

**Solution-**

import java.util.\*;

class CircularPrime

{

    boolean isPrime(int n) // Function for checking whether a number is prime or not

    {

        int c = 0;

        for(int i = 1; i<=n; i++)

        {

            if(n%i == 0)

                c++;

        }

        if(c == 2)

            return true;

        else

            return false;

    }

    int circulate(int n) //Function for circulating the digits to form new number

    {

        String s = Integer.toString(n);

        String p = s.substring(1)+s.charAt(0);

        int a = Integer.parseInt(p);

        return a;

    }

    void isCircularPrime(int n) //Function to check for circular prime

    {

        int f = 0,a = n;

        do

        {

            System.out.println(a);

            if(isPrime(a)==false)

            {

                f = 1;

                break;

            }

            a = circulate(a);

        }while(a!=n);

         if(f==1)

            System.out.println(n+" IS NOT A CIRCULAR PRIME");

        else

            System.out.println(n+" IS A CIRCULAR PRIME");

    }

         public static void main(String args[])

    {

        CircularPrime ob = new CircularPrime();

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter a number : ");

        int n = sc.nextInt();

        ob.isCircularPrime(n);

    }

}

**Output-**

Enter a number : 123  
123  
231  
312  
123 IS NOT A CIRCULAR PRIME

**Program 35**

A class Admission contain the admission numbers of 100 students. Some of the data members/ member functions are given below:

**Class name:** Admission

**Data member/instance variable:**

Adno[ ]: Integer array to store admission numbers

**Member functions/methods:**

Admission(): constructor to initialize the array elements  
void fillArray(): to accept the element of the array in ascending order  
int binSearch(int l, int u, int v): to search for a particular admission number(v) using binary search and recursive technique and return 1 if found otherwise returns -1

Specify the class **Admission** giving details of the **constructor**, **void fillArrray()** and **int binSearch(int, int, int)**. Define the **main()** function to create an object and call the functions accordingly to enable task.

**Solution-**

import java.util.\*;

class Admission

{

    int Adno[]=new int[100];

    static Scanner sc = new Scanner(System.in);

    Admission() // Default constructor

    {

        for(int i=0; i<100; i++)

        {

            Adno[i]=0;

        }

    }

    void fillArray()throws Exception // Function to accept elements in ascending order

    {

        for(int i=0; i<100; i++)

        {

            System.out.print("Enter Admission no of student "+(i+1)+": ");

            Adno[i] = sc.nextInt();

        }

        /\*Sorting the array in ascending order \*/

        int temp=0;

        for(int i=0; i<99; i++)

        {

            for(int j=i+1; j<100; j++)

            {

                if(Adno[i]>Adno[j])

                {

                    temp = Adno[i];

                    Adno[i] = Adno[j];

                    Adno[j] = temp;

                }

            }

        }

    }

    int binSearch(int l, int u, int v) // Recursive function implementing binary search

    {

        int mid = (l + u)/2;

        if(u < l) // condition if the search is unsuccessful

        {

            return -1;

        }

        if(v==Adno[mid]) // condition if the search is successful

        {

            return 1;

        }

        else if(v>Adno[mid])

        {

            return binSearch(mid+1,u,v);

        }

        else

        {

            return binSearch(l,mid-1,v);

        }

    }

    public static void main(String args[])throws Exception

    {

        Admission ob = new Admission();

        System.out.println("Enter Admission number in ascending order");

        ob.fillArray();

        System.out.print("Enter an Admission number to search : ");

        int v = sc.nextInt();

        int f = ob.binSearch(0,99,v);

        System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

        if(f == 1)

        {

            System.out.println("Admission Number found");

        }

        else

        {

            System.out.println("Admission Number Not found");

        }

    }

}

**Output-**

Enter Admission number in ascending order  
Enter Admission no of student 1: 205  
Enter Admission no of student 2: 310  
Enter Admission no of student 3: 670  
Enter Admission no of student 4: 887  
Enter Admission no of student 5: 952  
Enter an Admission number to search : 887  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
Admission Number found

**Program 36**

Write a program to input a word from the user and remove the duplicate characters present in it.

**Example:**

INPUT – abcabcabc  
OUTPUT – abc

**Solution-**

import java.io.\*;

class RemoveDupChar

{

    public static void main(String args[])throws IOException

    {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        System.out.print("Enter any word : ");

        String s = br.readLine();

        int l = s.length();

        char ch;

        String ans="";

        for(int i=0; i<l; i++)

        {

            ch = s.charAt(i);

            if(ch!=' ')

                ans = ans + ch;

            s = s.replace(ch,' '); //Replacing all occurrence of the current character by a space

        }

       System.out.println("Word after removing duplicate characters : " + ans);

    }

}

**Output-**

Enter any word : Mississippi  
Word after removing duplicate characters : Misp

**Program 37**

Design a class FiboString to generate Fibonacci strings. Some of the members of the class are given below:

Class name         :               FiboString

Data members/instance variables:

x                              :               to store the first string  
y                              :               to store the second string  
z                              :               to store the concatenation of the previous two strings  
n                             :               to store the number of terms

Member functions/methods:

FiboString()        :               constructor to assign x=”a”, y=”b”, z=”ba”  
void accept()      :               to accept the number of terms ‘n’  
void generate()   :               to generate and print the fibonacci strings. The sum of (‘+’ i.e. concatenation) first two strings is the third string. Eg. “a” is first string, “b” is second string then the third string will be “ba” and fourth will be “bab” and so on.

Specify the class FiboString, giving details of the constructor(), void accept() and void generate(). Define the main() function to create an object and call the functions accordingly to enable the task.

**Solution-**

import java.io.\*;

class FiboString

{

    String x,y,z;

    int n;

    FiboString() // Constructor

    {

        x = "a";

        y = "b";

        z = "ba"; // mentioned in the question otherwise not required. z = "" is sufficient

    }

    void accept()throws IOException

    {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        System.out.print("\nEnter the number of terms : ");

        n = Integer.parseInt(br.readLine());

    }

    void generate()

    {

        System.out.print("nThe Fibonacci String Series is : ");

        if(n <= 1) // If no of terms is less than or equal to 1

            System.out.print(x);

        else // If no of terms is more than or equal to 2

        {

            System.out.print(x+", "+y);

            for(int i=3; i<=n; i++)

            {

                z = y+x;

                System.out.print(", "+z);

                x = y;

                y = z;

            }

        }

    }

    public static void main(String args[]) throws IOException

    {

        FiboString ob = new FiboString();

        ob.accept();

        ob.generate();

    }

}

**Output-**

1) Enter the number of terms : 2  
The Fibonacci String Series is : a, b

2) Enter the number of terms : 5  
The Fibonacci String Series is : a, b, ba, bab, babba

**Program 38**

A class Mixer has been defined to merge two sorted integer arrays in ascending order. Some of the members of the class are given below:

**Class name                         :**Mixer

**Data members/instance variables:**

int arr[]                                 :               to store the elements of an array  
int n                                      :              to store the size of the array

**Member functions:**

Mixer( int nn)                     :               constructor to assign n = nn  
void accept()                     :               to accept the elements of the array in ascending order without any duplicates  
Mixer mix( Mixer A)             :              to merge the current object array elements with the parameterized array elements and return the resultant object  
void display()                    :               to display the elements of the array

Specify the class **Mixer**, giving details of the **constructor(int)**, **void accept()**, **Mixer mix(Mixer)** and**void display()**. Define the **main()** function to create an object and call the function accordingly to enable the task.

**Solution-**

import java.io.\*;

class Mixer

{

    int arr[];

    int n;

    static BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

    Mixer(int nn)

    {

        n = nn;

        arr = new int[n];

    }

    void accept()throws IOException

    {

        System.out.println("\n\* Input the Array \*\n");

        for(int i=0; i<n; i++)

        {

            System.out.print("Enter Element ["+(i+1)+"] : ");

            arr[i] = Integer.parseInt(br.readLine());

        }

        System.out.println();

    }

    Mixer mix(Mixer A)

    {

        int size = this.arr.length + A.arr.length; //size of resulting array

        Mixer B = new Mixer(size); //object which will store the result of merging

        int x = 0;

        /\* Merging the array of current object with array of parameter object \*/

        for(int i=0; i<size; i++)

        {

            if(i<A.arr.length)

                B.arr[i] = A.arr[i];

            else

            {

                B.arr[i] = this.arr[x];

                x++;

            }

        }

        /\* Sorting the result\*/

        int temp=0;

        for(int i=0; i<size-1; i++)

        {

            for(int j=i+1; j<size; j++)

            {

                if(B.arr[i]>B.arr[j])

                {

                    temp = B.arr[i];

                    B.arr[i] = B.arr[j];

                    B.arr[j] = temp;

                }

            }

        }

        return B;

    }

    void display()

    {

        for(int i=0; i<n; i++)

        {

            System.out.print(arr[i] + " ");

        }

        System.out.println();

    }

    public static void main(String args[])throws IOException

    {

        System.out.print("Enter size of the 1st array : ");

        int p = Integer.parseInt(br.readLine());

        Mixer obj1 = new Mixer(p);

        obj1.accept();

        System.out.print("Enter size of the 2nd array : ");

        int q = Integer.parseInt(br.readLine());

        Mixer obj2 = new Mixer(q);

        obj2.accept();

        Mixer obj3 = obj2.mix(obj1);

         System.out.print("The 1st Array is : ");

        obj1.display();

        System.out.print("The 2nd Array is : ");

        obj2.display();

        System.out.print("The Merged Array is : ");

        obj3.display();

    }

}

**Output-**

Enter size of the 1st array : 5

\* Input the Array \*

Enter Element [1] : 11  
Enter Element [2] : 13  
Enter Element [3] : 17  
Enter Element [4] : 24  
Enter Element [5] : 33

Enter size of the 2nd array : 3

\* Input the Array \*

Enter Element [1] : 5  
Enter Element [2] : 19  
Enter Element [3] : 30

The 1st Array is : 11 13 17 24 33  
The 2nd Array is : 5 19 30  
The Merged Array is : 5 11 13 17 19 24 30 33

**Program 39**

A super class Perimeter has been defined to calculate the perimeter of a parallelogram. Define a subclass Area to compute the area of the parallelogram by using the required data members of the super class. The details are given below:

Class name : Perimeter

Data members/instance variables:  
a : to store the length in decimal  
b : to store the breadth in decimal

Member functions:  
Perimeter( … ) : parameterized constructor to assign values to data members  
double Calculate( ) : calculate and return the perimeter of a parallelogram as 2 \* (length + breadth)  
void show() : to display the data members along with the perimeter of the parallelogram

Class name : Area

Data members/instance variables:  
h : to store the height in decimal  
area : to store the area of the parallelogram

Member functions:  
Area( … ) : parameterized constructor to assign values to data members of both the classes  
void doarea( ) : compute the area as (breadth \* height)  
void show() : display the data members of both classes along with the area and perimeter of the parallelogram.

**Solution-**

import java.io.\*;

class Perimeter //superclass

    {

        double a,b;

        Perimeter(double x, double y)

        {

            a=x;

            b=y;

        }

        double Calculate()

        {

            double p=2\*(a+b);

            return p;

        }

        void show()

        {

            System.out.println("Length = "+a);

            System.out.println("Breadth = "+b);

            System.out.println("Perimeter = "+Calculate()); //printing perimeter by calling Calculate()

        }

    } //end of superclass Perimeter

class Area extends Perimeter //subclass

    {

        double h, area;

        Area(double x, double y, double z)

        {

            super(x,y); //initializing data members of superclass by calling its constructor

            h=z;

        }

        void doarea()

        {

            area=b\*h;

        }

        void show()

        {

            super.show(); //calling the superclass function show()

            System.out.println("Height = "+h);

            System.out.println("Area = "+area);

        }

    } //end of subclass Area

public class Call //Class which will contain the main() method and execute the

    {

        public static void main(String args[])throws IOException

        {

            BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

            System.out.print("Enter The Length : ");

            double x=Double.parseDouble(br.readLine());

            System.out.print("Enter The Breadth : ");

            double y=Double.parseDouble(br.readLine());

            System.out.print("Enter The Height : ");

            double z=Double.parseDouble(br.readLine());

            Area ob=new Area(x,y,z); //creating object of subclass

            System.out.println("\*\*\* Output \*\*\*");

            ob.doarea();

            ob.show(); //calling show() function of subclass

        }

    }

**Output-**

Enter The Length : 5  
Enter The Breadth : 7  
Enter The Height : 9  
\*\*\* Output \*\*\*  
Length = 5.0  
Breadth = 7.0  
Perimeter = 24.0  
Height = 9.0  
Area = 63.0

**Program 40**

A class Recursion has been defined to find the Fibonacci series upto a limit. Some of the members of the class are given below:

Class Name : Recursion

Data Members/instance variables : a, b, c, limit (all integers)

Member functions/methods :

Recursion() : constructor to assign a,b,c with appropriate values.  
void input() : to accept the limit of the series.  
int fib(int n) : to return the nth Fibonacci term using recursive technique.  
void genearate\_fibseries() : to generate the Fibonacci series upto the given limit.

**Solution-**

import java.io.\*;

class Recursion

{

    static BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

    int a,b,c,limit;

Recursion() //Constructor

    {

     a=0;

     b=1;

     c=0;

     limit=0;

    }

void input()throws IOException //Function to input the limit

    {

     System.out.print("Enter the limit : ");

     limit=Integer.parseInt(br.readLine());

    }

int fib(int n) //Recursive function generating the 'nth' term of Fibonacci Series

    {

    if(n<=1)

    return a;

    else if(n==2)

    return b;

    else

    return (fib(n-1)+fib(n-2));

    }

 void generate\_fibseries()

    {

        System.out.println("The Fibonacci Series is:");

        for(int i=1;i<=limit;i++)

        {

            c=fib(i);

            System.out.print(c+"  ");

        }

    }

 public static void main(String args[])throws IOException

  {

   Recursion ob=new Recursion();

   ob.input();

   ob.generate\_fibseries();

  }

}

**Output-**

Enter the limit : 11  
The Fibonacci Series is:  
0 1 1 2 3 5 8 13 21 34 55

Enter the limit : 20  
The Fibonacci Series is:  
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181